

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-103104

(43)Date of publication of application : 15.04.1994

(51)Int.Cl.

G06F 11/28

G06F 9/06

G06F 15/16

(21)Application number : 04-249296

(71)Applicant : HITACHI LTD  
HITACHI ASAHI ELECTRON:KK

(22)Date of filing : 18.09.1992

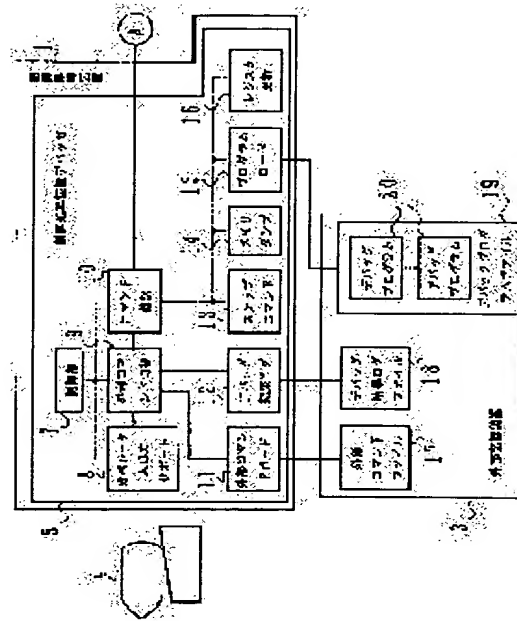
(72)Inventor : HOSHINA KIYOSHI  
UCHIKAWA YOSHIHIKO  
HAGIWARA TOSHIO  
TAKAGI TOSHIKI  
TAMURA HIDEKATSU

## (54) REMOTE DEBUG METHOD

### (57)Abstract:

**PURPOSE:** To use the same debugger of control processor for a processor to be controlled of every different kind by absorbing exclusively only an intrinsic part related to an architecture of the processor to be controlled, by the debugger of the processor to be controlled.

**CONSTITUTION:** In a control processor 1, a function of command translation 10 is provided, and the function of its command translation 10 receives an external command from an operator, or an external command file 17 through a function of external command switching 9, translates its command combination of an internal command, and transfers successively the internal command to a processor to be controlled corresponding to a device number added to the external command. The processor to be controlled executes successively the internal command by a processing peculiar to the device. In such a state, the operator checks a debug result log file 18, can know a result of debug, and can execute program debug of the processor to be controlled.



## LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

JP,06-103104,A

\* NOTICES \*

- 5 JPO and NCIP are not responsible for any damages caused by the use of this translation.
1. This document has been translated by computer. So the translation may not reflect the original precisely.
  2. \*\*\*\* shows the word which can not be translated.
  3. In the drawings, any words are not translated.

10

-----  
CLAIMS  
-----

15 [Claim(s)]

[Claim 1] The remote debugging approach characterized by to combine two or more sets of processors, to translate the external command which inputs the command for debugging of an operator into said control processor at said control processor in the remote debugging approach which debugs the program of the controlled processor which are other processors from the control processor which is one set of said processor into the combination of the internal command which said controlled processor executes, and to transmit the combination of this internal command to said controlled processor.

[Claim 2] Said external command is parameterized as an input of said external command. It stores in the storage of said control processor as an external command file beforehand. The external command file load approach which loads the file concerned is made possible. Moreover, it has the debugging result log function which records a debugging result on the debugging result log file of the store of said control processor. The remote debugging approach according to claim 1 characterized by considering whether an external command is received from an operator, or the parameter of an external command is received from said external command file as the external command change to change.

[Claim 3] The remote debugging approach given in either of claims 1 or 2 characterized by performing debugging of two or more sets of controlled processors by preparing the device number in said controlled processor correspondence, performing the parameter of said external command, and control of a debugging result log file based on the device number of said controlled processor set as said parameter and log file, and specifying said parameter and log file as the controlled processor corresponding to said device number.

-----  
40 DETAILED DESCRIPTION  
-----

[Detailed Description of the Invention]

[0001]

[Industrial Application] In order that especially this invention may give versatility about the remote debugging approach of a program, By making a controlled processor take charge of only processing of dedication, and making a control processor take charge of all the other processings The debugger of a control processor can be used in common, and debugging of a program can be performed automatically, and debugging of the controlled processor with which two or more sets of architecture differed is further related with the suitable remote debugging approach to raise the debugging effectiveness whose activation was enabled to coincidence.

[0002]

[Description of the Prior Art] The former, for example, JP,55-121538,A, Like remote control of the computer indicated by the official report, two sets of processors are connected, a man machine interface is given to one control processor, and another processor is controlled, and although the approach an operator operates it with the controlled processor concerned is learned, it is not taken into consideration especially about the problem of the functional assignment of a control processor and a controlled processor here. Moreover, it is not taken into consideration about the problem of the debugging effectiveness by full automation, either. Moreover, although, as for the case of the processor which used the microprocessor, an incircuit emulator (henceforth "ICE") is generally used, when this demounts a microprocessor, attaches PURUBU of ICE instead, and emulates it by the equivalent microprocessor by the ICE side and the class of microprocessor for debugging changes, there is a problem that it is necessary to change an ICE body and PURUBU corresponding to it. Moreover, in the case of ICE, the delay of the processing speed by emulating the microprocessor concerned poses a problem.

[0003]

[Problem(s) to be Solved by the Invention] Especially the above-mentioned conventional technique is not taken into consideration about the versatility corresponding to the class from which the controlled processor which processing of a control processor supports differed. Moreover, by the approach using ICE, there is a problem that it is necessary to prepare target ICE body or PURUBU which corresponds for every microprocessor. Moreover, when ICE is connected, there is also another problem of differing from the processing speed of actual equipment. The place which this invention was made in view of the above-mentioned situation, and is made into the purpose is to offer the remote debugging approach without fluctuation of processing speed in operating on equipment with the actual program which solves the problem like \*\*\*\* in a Prior art, and can use the debugger of the same control processor corresponding to the controlled processor for every different class, and is debugged. Moreover, other purposes of this invention are by registering the external command of debugging into the file of a control processor beforehand, carrying out sequential execution of the external command on an above-mentioned file instead of specified by an operator, and recording the activation result on the file to offer the approach of enabling automatic debugging. Furthermore, in order that a control processor may not be dependent on the class of controlled processor, an above-mentioned external command is registered beforehand and it is in offering automatically

the remote DEBA@GGU approach which can be performed about debugging of the program of the controlled processor with which two or more sets of classes differed.

[0004]

[Means for Solving the Problem] In the remote debugging approach which the above-mentioned purpose of this invention combines two or more sets of processors, and debugs the program of the controlled processor which are other processors from the control processor which is one set of said processor To said control processor, the external command which inputs the command for debugging of an operator into said control processor It translates into the combination of the internal command which said controlled processor executes, and is attained by the remote debugging approach characterized by transmitting the combination of this internal command to said controlled processor.

[0005]

[Function] In the remote debugging approach concerning this invention, a command translation function is prepared in a control processor, and through an external command change function, an external command is translated into reception from an operator or an external command file, it translates the command concerned into the combination of an internal command, and this command translation function carries out the sequential transfer of the internal command at the controlled processor corresponding to the device number added to the external command. A controlled processor executes the received internal command by processing peculiar to sequential equipment. Moreover, in order that the program to debug may operate on a controlled processor, the problem on processing speed like [ in the case of using ICE ] is not generated. Moreover, it becomes possible to perform debugging automatically for the information read in the external command file by preparing the external command change function which changes whether it is based on the external command file which registers an external command beforehand, the debugging result log file which records a debugging result, and whether a debug command is depended on an operator's input and said external command file in a control processor. In addition, an internal command can be transmitted to the controlled processor concerned making the connection address of a controlled processor correspond with the device number, and by specifying the device number of a controlled processor in said external command. Moreover, since as a result of [ of which controlled processor ] debugging it is can distinguish by putting in the device number [ be / it / under / of the information on a debugging result / correspondence ], it becomes possible to debug to coincidence to two or more sets of controlled processors.

[0006]

[Example] Hereafter, the example of this invention is explained to a detail based on a drawing. The program-block Fig. of the remote debugger which drawing 1 and drawing 2 show one example of this invention, and drawing 3 are the program-block Figs. of the command translation processing section 10 in drawing 1 . In drawing 1 and drawing 2 , in 1, a controlled processor and 3 show the external storage of the control processor 1, and, as for a control processor and 2, 4 shows the operator I/O device. 5 a controlled processor debugger and 7 for a control processor debugger and 6 Moreover, a control section, The operator I/O support

processing section and 9 8 The external command change processing section, 10 the external command load processing section and 12 for the command translation processing section and 11 The debugging result log processing section, 13 the memory-dump processing section and 15 for the step command-processing section and 14 The program load processing section, 16 an external command file and 18 for the register update process section and 17 A debugging result log file, 19 a debug program and 21 for a debug program file and 20 The internal-command processing section, 22 -- the memory read/write processing section and 23 -- the register read/write processing section and 24 -- in the interrupt-processing section and 25, Address A and 28 show the address (A+1), and, as for the instruction check processing section and 26, 29 shows Address B, as for main storage and 27.

[0007] In drawing 3 30 a managed table and 32 for command translation processing and 31 The device number of the controlled processor 2, The table name for external command conversion corresponding to the controlled processor 2 in 33, The equipment information table name corresponding to the controlled processor 2 and 35 34 A managed table end mark, 36 an external command and 38 for the table for external command conversion, and 37 An internal command, 39 -- the end mark of an internal command, and 40 -- in the end mark of the table for external command conversion, and 41, register information and 44 show special memory information, and, as for an equipment information table and 42, 45 shows the end mark of an equipment information table, as for memory information and 43. When an operator debugs, the external command change processing section 9 requires the input from an operator of the operator I/O support processing section 8. The operator I/O support processing section 8 performs the display of the waiting for an external command input to an operator to operator I/O device 4, and waits for an operator's input. An operator inputs the program load of the external command which loads the program to debug from operator I/O device 4. The information given and inputted into the program load of an external command is the memory address of the program name to debug and the controlled processor 2 which loads a program, the device number of the controlled processor 2, etc.

[0008] The operator I/O support processing section 8 passes an external command, a debug program name, a memory address, the device number, etc. to the external command change processing section 9. The external command change processing section 9 passes the above-mentioned information to the command translation processing section 10. Search the command translation processing section 10 with the device number in [ out of the device number 32 of the controlled processor registered into the managed table 31 ] the above-mentioned information by the command translation processing 30, and in the congruous places The table name 33 for external command conversion corresponding to the controlled processor 2 and the equipment information table name 34 corresponding to the controlled processor 2 are given to the above-mentioned information, and control is passed to the program load processing section 15 which actually carries out translation activation of the external command concerned. The program load processing section 15 acquires the memory light command which is an internal command 38 in the place which searched the external command 37 in the table 36 for external command conversion with the external command of an operator

input based on the table name 33 for external command conversion corresponding to the received controlled processor 2, and was in agreement with it. Moreover, the program load processing section 15 decides that the memory address of the controlled processor 2 which is the memory information 42 in the equipment information table 41, and an operator input loads based on the equipment information table name 34 corresponding to the received controlled processor 2 is proper, and acquires the access smallest unit to the memory of the controlled processor 2 etc. further.

[0009] Furthermore, the program load processing section 15 discovers the program name which an operator input debugs from the debug program file 19, and reads a debug program 20 on the primary storage of the control processor 1. It adds, and translation edit is carried out and the information that the proper thing decided the read debug program 20 to the predetermined number of transfer data based on the access smallest unit to the memory of the above-mentioned controlled processor 2, such as a memory address of the controlled processor 2 to load, is transmitted to memory light command assignment of the acquired internal command at the controlled processor 2 of the specified device number. In the controlled processor 2, the internal-command processing section 21 passes said internal command to reception and the memory read/write processing section 22. The memory read/write processing section 22 writes the number of the data in main storage 26 from the memory address specified by the internal command. The program load processing section 15 loads a debug program 20, and repeats the transfer by memory light command assignment until the writing to the main storage 26 of the device number with which the debug program 20 was specified is completed.

[0010] If the writing to the main storage 26 of the device number with which the debug program 20 was specified is completed, the program load processing section 15 will be the table 36 for external command conversion, and will acquire the following internal command. When it is the end mark 39 of an internal command, the program load processing section 15 connects completion of a command to the external command change processing section 9 through the command translation processing section 10. The external command change processing section 9 connects completion of an external command to the operator I/O support processing section 8. The operator I/O support processing section 8 displays that the program load was completed on operator I/O device 4, and serves as input waiting of a command. Moreover, when an operator checks a memory content, it is the external command of memory dump, and if range, the device number, etc. of a memory address of the controlled processor 2 are specified, the command translation processing section 10 will pass control like the above to the memory-dump processing section 14. The memory-dump processing section 14 adds information which the proper thing decided, such as a memory address and the predetermined number of transfer data, carries out translation edit and transmits it to memory lead command assignment of the acquired internal command at the controlled processor 2 of the specified device number.

[0011] In the controlled processor 2, the internal-command processing section 21 passes said internal command to reception and the memory read/write processing section 22. From the memory address specified by the internal command, the memory read/write processing section 22 reads the number of the data from main storage 26, and transmits it to the control processor

1. The memory-dump processing section 14 repeats this processing to the address specified with the external command. If a memory content is received to the address specified with the external command, the memory-dump processing section 14 will edit a memory content into a hexadecimal display etc., and will pass delivery and the external command change processing section 9 to the external command change processing section 9 through [ ten ] the command translation processing section at the operator I/O support processing section 8. The operator I/O support processing section 8 displays the received information on operator I/O device 4. Moreover, by the same approach, by the command translation processing section 10 and the register update process section 16, the writing of the contents of a register and a display are also translated into an internal command from an external command, and are performed through the register read/write processing section 23.

[0012] Next, from address [ of a program ] A (27), an operator makes an example the case where a condition is checked, performing for every step, and explains more concretely. An operator specifies the step command of an external command like [ this ] the above-mentioned processing. A step command is passed to the command translation processing section 10 through the operator I/O support processing section 8 and the external command change processing section 9, and the command translation processing section 10 passes control to the step command-processing section 13. The step command-processing section 13 adds information, such as a register value required for the display after the address and activation, and performs the exchange of the controlled processor 2 and information in the following procedures.

\*\* The step command-processing section 13 transmits the internal command of memory lead command assignment of an address (A+1) in order to secure the contents of the address (A+1) address (28).

\*\* The internal-command processing section 21 of the controlled processor 2 makes the memory read/write processing section 22 read the contents of the address (A+1), and transmit it to control processor side 1.

\*\* The step command-processing section 13 transmits the memory light command of an internal command to the controlled processor 2 so that the instruction which causes program interruption may next be written in an address (A+1).

[0013] \*\* The internal-command processing section 21 of the controlled processor 2 writes said instruction in an address through the memory read/write processing section 22 (A+1).

\*\* Next, the step command-processing section 13 transmits register light command assignment of an internal command so that a program counter may be set to the Ath address (27).

\*\* The internal-command processing section 21 of the controlled processor 2 sets the Ath address to a program counter through the register read/write processing section 22. If an address (A+1) address is performed after Ath address activation with the set of a program counter, program interruption will occur and control will include the interrupt-processing section 24 of the controlled processor 2. The interrupt-processing section 24 performs interrupt processing of evacuating a register, and passes control to the internal-command processing section 21. The internal-command processing section 21 adds that the interrupt occurred, and



transmits interruption information, such as a register, for it to the control processor 1.

\*\* If it checks that it is the information from the controlled processor 2, and it is the interruption which interruption expected, the step command-processing section 13 of the control processor 1 will make it the controlled processor 2 to require a transfer etc., if it has the contents of a register, and still more nearly required information, it will return the contents of the secured address ( A+1) address to the controlled processor 2 by the internal command of memory light command assignment after that, and will finish 1 step executive operation.

[0014] \*\* After displaying the step command-processing section 13 on an operator through the command translation processing section 10, the external command change processing section 9, and the operator I/O support processing section 8, it waits for a check of an operator. When an operator checks and it continues activation of the following one step, the next address is specified and the same processing is repeated. In addition, although above-mentioned explanation explains supposing a run command not being branch instruction, since the support of branch instruction is based on the instruction of the controlled processor 2, it is absorbed with the controlled processor 2. For example, in advance of activation of the aforementioned \*\*, the step command-processing section transmits the internal command of the instruction check processing section which checks the Ath contents, and the internal-command processing section 21 of the controlled processor 2 judges whether it is branch instruction in the instruction check processing section 25, and transmits the address immediately after the instruction concerned, and the branching place address, for example, the Bth street. The step command-processing section 13 secures the contents of both branching places immediately after a run command, and writes in the instruction which causes program interruption to both, the instruction which branches in the internal-command processing section 21 which is the common return point.

[0015] Moreover, the instruction which causes said program interruption is prepared with the controlled processor 2, and the internal command which writes in the instruction into which the flow of the instruction execution of the controlled processor 2 is changed is transmitted from the control processor 1. Here, in order to simplify explanation, an instruction length is a part for the 1st street, and is premised on not being branch instruction. Moreover, the external command is beforehand stored as the parameter formal debugging procedure at the external command file 17, for example, when performing automatic debugging from the external command file 17, at the time of loading of the control processor debugger 5, it carries out giving an automatic debug mode with a load parameter etc., and after loading, the external command change processing section 9 reads said mode in specific area etc., and is taken as an automatic debug mode. The external command change processing section 9 requires an external command of the external command load processing section 11 instead of requiring an external command of the operator I/O support processing section 8. The external command load processing section 11 reads an external command from the external command file 17 one by one, and passes it to the external command change processing section 9.

[0016] The external command change processing section 9 passes an external command to the command translation processing section 10 like the input from the operator I/O support

processing section 8. If the activation result of an external command is sent from the command translation processing section 10 after activation termination of an external command, in the external command change processing section 9, the information on an activation result is stored in the debugging result log processing section 12, and delivery and the debugging result log processing section 12 store this in the debugging result log file 18. As mentioned above, debugging can be automatically performed like the case where an operator directs. After all debugging is completed, an operator can check the debugging result log file 18, and can know a debugging result. According to the above-mentioned example, program debugging of the controlled processor 2 can be performed in this way, and the versatility of the debugger program of the control processor 1 is raised, and the laborsaving by automation of debugging can be attained. Moreover, two or more sets of the controlled processors 2 can be debugged to coincidence by one set of the control processor 1. In addition, since the item debugged before can be saved with the parameter of an external command, the next leakage in debugging can be prevented.

[0017]

[Effect of the Invention] As mentioned above, as explained to the detail, according to this invention, the remarkable effectiveness that the remote debugging approach without fluctuation of processing speed is realizable in operating on equipment with the actual program which can use the debugger of the same control processor corresponding to the controlled processor for every different class, and is debugged is done so. In a detail, versatility can be given to a control processor debugger by absorbing only the part of the proper in connection with the architecture of a controlled processor to dedication with a controlled processor debugger, and if it gives only a control processor debugger, a functional addition etc. can be adapted for this whole debugging system, and its expandability is also more high. Furthermore, automation of debugging can be measured by parameterizing an external command to an external command file, and storing as the debugging procedure. Moreover, since the item which always debugged before can be repeated and performed by accumulating the parameter of an external command to an external command file, the degradation of a program is prevented and reliability can be maintained. Moreover, debugging of the program from which the controlled processor with which two or more sets of architecture differed differed can perform automatically by setting the device number of a controlled processor as an external command.

[0018]

## DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] They are some program-block Figs. of the remote debugger in which one example of this invention is shown.

[Drawing 2] They are some other program-block Figs. of the remote debugger in which one

example of this invention is shown.

[Drawing 3] It is the program-block Fig. of the command translation processing section 10 in drawing 1 .

[Description of Notations]

- 5 A control processor, the processor controlled [ 2:], 3 : 1: The external storage of a control processor, 4: An operator I/O device, 5:control processor debugger, 6 : A controlled processor debugger, 7 : A control section, 8:operator I/O support processing section, 9:external command change processing section, 10: The command translation processing section, 11:external command load processing section, 12 : The debugging result log processing section, 13: The step
- 10 command-processing section, 14:memory-dump processing section, 15 : The program load processing section, 16: The register update process section, a 17:external command file, 18 : A debugging result log file, 19: A debug program file, 20:debug program, 21 : The internal-command processing section, 22: The memory read/write processing section, 23:register read/write processing section, 24:interrupt-processing section, 25:instruction check
- 15 processing section, 26:main storage, the 27:addresses A, 28:address (A+1), 29: Address B.

-----  
[Translation done.]

20

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平6-103104

(43)公開日 平成6年(1994)4月15日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 11/28  
9/06  
15/16

識別記号

A 9290-5B  
4 4 0 S 9367-5B  
4 5 0 Z 9190-5L

庁内整理番号

F I

技術表示箇所

審査請求 未請求 請求項の数3(全 8 頁)

(21)出願番号 特願平4-249296

(22)出願日 平成4年(1992)9月18日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(71)出願人 391002384

株式会社日立旭エレクトロニクス

愛知県尾張旭市晴丘町池上1番地

(72)発明者 保科 清

愛知県尾張旭市晴丘町池上1番地 株式会

社日立製作所旭工場内

(72)発明者 内川 芳彦

愛知県尾張旭市晴丘町池上1番地 株式会

社日立製作所旭工場内

(74)代理人 弁理士 磯村 雅俊

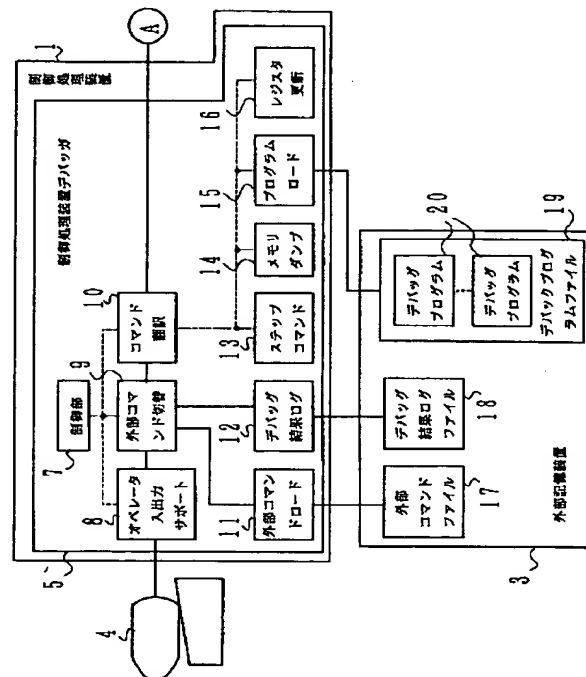
最終頁に続く

(54)【発明の名称】 リモートデバッグ方法

(57)【要約】

【目的】 異なった種類毎の被制御処理装置に対応して同一の制御処理装置のデバッガが使用でき、また、デバッグするプログラムが実際の装置上で動作することで処理速度の変動が無いリモートデバッグ方法を提供すること。

【構成】 複数台の処理装置を結合し、前記処理装置中の1台である制御処理装置から他の処理装置である被制御処理装置のプログラムをデバッグするリモートデバッグ方法において、前記制御処理装置に、オペレータがデバッグのためのコマンドを前記制御処理装置に入力する外部コマンドを、前記被制御処理装置が実行する内部コマンドの組合せに翻訳し、該内部コマンドの組合せを前記被制御処理装置に転送することを特徴とするリモートデバッグ方法。



## 【特許請求の範囲】

【請求項 1】 複数台の処理装置を結合し、前記処理装置中の 1 台である制御処理装置から他の処理装置である被制御処理装置のプログラムをデバッグするリモートデバッグ方法において、前記制御処理装置に、オペレータがデバッグのためのコマンドを前記制御処理装置に入力する外部コマンドを、前記被制御処理装置が実行する内部コマンドの組合せに翻訳し、該内部コマンドの組合せを前記被制御処理装置に転送することを特徴とするリモートデバッグ方法。

【請求項 2】 前記外部コマンドの入力として、前記外部コマンドをパラメータ化し、予め外部コマンドファイルとして前記制御処理装置の記憶装置に格納し、当該ファイルをロードする外部コマンドファイルロード方法を可能とし、また、デバッグ結果を前記制御処理装置の記憶装置のデバッグ結果ログファイルへ記録するデバッグ結果ログ機能を有し、オペレータから外部コマンドを受け取るか前記外部コマンドファイルから外部コマンドのパラメータを受け取るかを切り替える外部コマンド切替としたことを特徴とする請求項 1 記載のリモートデバッグ方法。

【請求項 3】 前記被制御処理装置対応に装置番号を設けて、前記外部コマンドのパラメータとデバッグ結果ログファイルの制御を、前記パラメータとログファイルに設定した前記被制御処理装置の装置番号に基づいて行い、前記パラメータとログファイルを前記装置番号に対応する被制御処理装置に特定することにより、複数台の被制御処理装置のデバッグを実行することを特徴とする請求項 1 または 2 のいずれかに記載のリモートデバッグ方法。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】 本発明はプログラムのリモートデバッグ方法に関し、特に汎用性を持たせるため、被制御処理装置には専用の処理のみを受け持たせ、それ以外の処理はすべて制御処理装置に受け持たせることにより、制御処理装置のデバッグを共通に使用でき、また、プログラムのデバッグを自動的に実行でき、更に、複数台のアーキテクチャの異なった被制御処理装置のデバッグが同時に実行可能とした、デバッグ効率を高めるのに好適なリモートデバッグ方法に関する。

## 【0002】

【従来の技術】 従来、例えば、特開昭55-121538号 公報に開示された電子計算機の遠隔制御装置のように、2 台の処理装置を接続し、一方の制御処理装置にマンマシンインタフェースを持たせてもう一方の処理装置を制御し、当該被制御処理装置で操作員が操作を行う方法は知られているが、ここでは、制御処理装置と被制御処理装置の機能分担の問題については特に考慮されていない。また、無人化によるデバッグ効率の問題についても考慮

されていない。また、一般的に、マイクロプロセッサを使用した処理装置の場合は、インサーキットエミュレータ(以下、「ICE」という)が使用されるが、これはマイクロプロセッサを取外し、代わりに ICE のブルーブを取付け ICE 側で同等のマイクロプロセッサでエミュレートするもので、デバッグ対象のマイクロプロセッサの種類が変わると、それに対応して、ICE 本体やブルーブを変えることが必要となるという問題がある。また、ICE の場合、当該マイクロプロセッサをエミュレートすることによる処理速度の遅れが問題となる。

## 【0003】

【発明が解決しようとする課題】 上記従来技術は、制御処理装置の処理の、特にサポートする被制御処理装置の異なった種類に対応する汎用性については、考慮されていない。また、ICE を用いる方法では、対象とするマイクロプロセッサ毎に、対応する ICE 本体またはブルーブを用意する必要があるという問題がある。また、ICE を接続した場合、実際の装置の処理速度と異なるという別の問題もある。本発明は上記事情に鑑みてなされたもので、その目的とするところは、従来の技術における上述の如き問題を解消し、異なった種類毎の被制御処理装置に対応して同一の制御処理装置のデバッグが使用でき、また、デバッグするプログラムが実際の装置上で動作することで処理速度の変動が無いリモートデバッグ方法を提供することにある。また、本発明の他の目的は、予めデバッグの外部コマンドを制御処理装置のファイルに登録しておき、オペレータ指定の代わりに上述のファイル上の外部コマンドを順次実行し、実行結果をファイルに記録しておくことにより、自動的なデバッグを可能とする方法を提供することにある。更に、制御処理装置が被制御処理装置の種類に依存しないため、予め上述の外部コマンドを登録し、複数台の種類の異なった被制御処理装置のプログラムのデバッグを、自動的に実行可能なりモートデバッグ方法を提供することにある。

## 【0004】

【課題を解決するための手段】 本発明の上記目的は、複数台の処理装置を結合し、前記処理装置中の 1 台である制御処理装置から他の処理装置である被制御処理装置のプログラムをデバッグするリモートデバッグ方法において、前記制御処理装置に、オペレータがデバッグのためのコマンドを前記制御処理装置に入力する外部コマンドを、前記被制御処理装置が実行する内部コマンドの組合せに翻訳し、該内部コマンドの組合せを前記被制御処理装置に転送することを特徴とするリモートデバッグ方法によって達成される。

## 【0005】

【作用】 本発明に係るリモートデバッグ方法においては、制御処理装置にコマンド翻訳機能を設け、該コマンド翻訳機能は、外部コマンド切替機能を介し、オペレータまたは外部コマンドファイルから外部コマンドを受け

取り、当該コマンドを内部コマンドの組合せに翻訳し、外部コマンドに付加された装置番号に対応する被制御処理装置に内部コマンドを順次転送する。被制御処理装置は受け取った内部コマンドを順次装置特有の処理で実行する。また、デバッグするプログラムは被制御処理装置上で動作するため、ICEを使用する場合のような処理速度上の問題は発生しない。また、予め外部コマンドを登録する外部コマンドファイルとデバッグ結果を記録するデバッグ結果ログファイルと、デバッグコマンドをオペレータの入力による前記外部コマンドファイルによるかを切り替える外部コマンド切替機能を制御処理装置に設けることにより、外部コマンドファイルから読み取った情報で自動的にデバッグを実行することが可能になる。なお、被制御処理装置の接続アドレスを装置番号と対応させることと、前記外部コマンドの中に被制御処理装置の装置番号を指定することにより、内部コマンドを当該被制御処理装置に転送できる。また、デバッグ結果の情報の中に対応する装置番号を入れることにより、どの被制御処理装置のデバッグ結果かが判別できるため、複数台の被制御処理装置に対して同時にデバッグすることが可能になる。

#### 【0006】

【実施例】以下、本発明の実施例を図面に基づいて詳細に説明する。図1および図2は、本発明の一実施例を示すリモートデバッグのプログラムブロック図、図3は、図1中のコマンド翻訳処理部10のプログラムブロック図である。図1および図2において、1は制御処理装置、2は被制御処理装置、3は制御処理装置1の外部記憶装置、4はオペレータ入出力装置を示している。また、5は制御処理装置デバッグ、6は被制御処理装置デバッグ、7は制御部、8はオペレータ入出力サポート処理部、9は外部コマンド切替処理部、10はコマンド翻訳処理部、11は外部コマンドロード処理部、12はデバッグ結果ログ処理部、13はステップコマンド処理部、14はメモリダンプ処理部、15はプログラムロード処理部、16はレジスタ更新処理部、17は外部コマンドファイル、18はデバッグ結果ログファイル、19はデバッグプログラムファイル、20はデバッグプログラム、21は内部コマンド処理部、22はメモリリードライト処理部、23はレジスタリードライト処理部、24は割込み処理部、25は命令チェック処理部、26は主記憶装置、27はアドレスA、28はアドレス(A+1)、29はアドレスBを示している。

【0007】図3において、30はコマンド翻訳処理、31は管理テーブル、32は被制御処理装置2の装置番号、33は被制御処理装置2に対応した外部コマンド変換用テーブル名称、34は被制御処理装置2に対応した装置情報テーブル名称、35は管理テーブル終了マーク、36は外部コマンド変換用テーブル、37は外部コマンド、38は内部コマンド、39は内部コマンドの終

了マーク、40は外部コマンド変換用テーブルの終了マーク、41は装置情報テーブル、42はメモリ情報、43はレジスタ情報、44は特殊メモリ情報、45は装置情報テーブルの終了マークを示している。オペレータがデバッグを行う場合、外部コマンド切替処理部9は、オペレータ入出力サポート処理部8にオペレータからの入力を要求する。オペレータ入出力サポート処理部8は、オペレータ入出力装置4にオペレータに対する外部コマンド入力待ちの表示を行い、オペレータの入力を待つ。オペレータは、デバッグするプログラムをロードする外部コマンドのプログラムロードを、オペレータ入出力装置4から入力する。外部コマンドのプログラムロードにつけて入力する情報は、デバッグするプログラム名称や、プログラムをロードする被制御処理装置2のメモリアドレスや被制御処理装置2の装置番号等である。

【0008】オペレータ入出力サポート処理部8は、外部コマンドとデバッグプログラム名称とメモリアドレスおよび装置番号等を、外部コマンド切替処理部9に渡す。外部コマンド切替処理部9は、上記情報を、コマンド翻訳処理部10に渡す。コマンド翻訳処理部10は、コマンド翻訳処理30により、管理テーブル31中に登録してある被制御処理装置の装置番号32中から、上記情報中の装置番号で検索し、一致したところで、被制御処理装置2に対応した外部コマンド変換用テーブル名称33と、被制御処理装置2に対応した装置情報テーブル名称34を前述の情報に付与して、実際に当該外部コマンドを翻訳実行するプログラムロード処理部15に制御を渡す。プログラムロード処理部15は、受け取った被制御処理装置2に対応した外部コマンド変換用テーブル名称33を基に、外部コマンド変換用テーブル36中の外部コマンド37を、オペレータ入力の外部コマンドで検索し、一致したところで内部コマンド38であるメモリライトコマンドを取得する。また、プログラムロード処理部15は、受け取った被制御処理装置2に対応した装置情報テーブル名称34を基に、装置情報テーブル41中のメモリ情報42で、オペレータ入力のロードする被制御処理装置2のメモリアドレスが適正であることを確定し、更に、被制御処理装置2のメモリへのアクセス最小単位等を取得する。

【0009】更に、プログラムロード処理部15は、オペレータ入力のデバッグするプログラム名称をデバッグプログラムファイル19から捜し出し、デバッグプログラム20を制御処理装置1の主記憶上に読み出す。読み出したデバッグプログラム20を、前述の被制御処理装置2のメモリへのアクセス最小単位を基に、所定の転送データ数にし、適正であることが確定したロードする被制御処理装置2のメモリアドレス等の情報を付加し、取得した内部コマンドのメモリライトコマンド指定に翻訳編集し、指定された装置番号の被制御処理装置2に転送する。被制御処理装置2では、内部コマンド処理部21

が前記内部コマンドを受け取り、メモリリードライト処理部22に渡す。メモリリードライト処理部22は、内部コマンドで指定されたメモリアドレスから指定データ数を主記憶装置26に書き込む。プログラムロード処理部15は、デバッグプログラム20の、指定された装置番号の主記憶装置26への書き込みが完了するまで、デバッグプログラム20をロードし、メモリライトコマンド指定での転送を繰り返す。

【0010】デバッグプログラム20の、指定された装置番号の主記憶装置26への書き込みが完了すると、プログラムロード処理部15は、外部コマンド変換用テーブル36で、次の内部コマンドを取得する。それが内部コマンドの終了マーク39である場合、プログラムロード処理部15は、コマンド翻訳処理部10を介して外部コマンド切替処理部9にコマンドの完了を連絡する。外部コマンド切替処理部9は、オペレータ入出力サポート処理部8に、外部コマンドの完了を連絡する。オペレータ入出力サポート処理部8は、オペレータ入出力装置4に、プログラムロードが完了したことを表示し、コマンドの入力待ちとなる。また、オペレータがメモリ内容をチェックする場合は、例えば、メモリダンプの外部コマンドで、被制御処理装置2のメモリアドレスの範囲と装置番号等を指定すると、前記と同様に、コマンド翻訳処理部10は、メモリダンプ処理部14に制御を渡す。メモリダンプ処理部14は、適正であることが確定したメモリアドレス、所定の転送データ数等の情報を付加し、取得した内部コマンドのメモリリードコマンド指定に翻訳編集し、指定された装置番号の被制御処理装置2に転送する。

【0011】被制御処理装置2では、内部コマンド処理部21が前記内部コマンドを受け取り、メモリリードライト処理部22に渡す。メモリリードライト処理部22は、内部コマンドで指定されたメモリアドレスから、指定データ数を主記憶装置26から読み、制御処理装置1に転送する。メモリダンプ処理部14は、この処理を外部コマンドで指定されたアドレスまで繰り返す。外部コマンドで指定されたアドレスまでメモリ内容を受け取ると、メモリダンプ処理部14は、メモリ内容を16進表示等に編集してコマンド翻訳処理部10を介し、外部コマンド切替処理部9に渡し、外部コマンド切替処理部9は、オペレータ入出力サポート処理部8に渡す。オペレータ入出力サポート処理部8は、受け取った情報をオペレータ入出力装置4に表示する。また、レジスタ内容の書き込み、表示も同様の方法で、コマンド翻訳処理部10とレジスタ更新処理部16とにより、外部コマンドから内部コマンドに翻訳して、レジスタリードライト処理部23を介して実行する。

【0012】次に、オペレータが、プログラムのアドレスA(27)から、1ステップごとに実行しながら状態をチェックする場合を例にして、より具体的に説明する。

これも前述の処理と同様に、オペレータは外部コマンドのステップコマンドを指定する。ステップコマンドは、オペレータ入出力サポート処理部8、外部コマンド切替処理部9を介してコマンド翻訳処理部10に渡され、コマンド翻訳処理部10は、ステップコマンド処理部13に制御を渡す。ステップコマンド処理部13は、アドレスと実行後の表示に必要なレジスタ値等の情報を付加し、次のような手順で被制御処理装置2と情報のやり取りを実行する。

①ステップコマンド処理部13は、アドレス(A+1)番地(28)の内容を確保するため、(A+1)番地のメモリリードコマンド指定の内部コマンドを転送する。

②被制御処理装置2の内部コマンド処理部21は、メモリリードライト処理部22に(A+1)番地の内容を読み取らせ、制御処理装置側1に転送する。

③ステップコマンド処理部13は、次に(A+1)番地にプログラム割込みを起こす命令を書き込むよう、内部コマンドのメモリライトコマンドを被制御処理装置2に転送する。

【0013】④被制御処理装置2の内部コマンド処理部21は、メモリリードライト処理部22を介し(A+1)番地に前記命令を書き込む。

⑤次に、ステップコマンド処理部13は、プログラムカウンタをアドレスA番地(27)にセットするよう、内部コマンドのレジスタライトコマンド指定を転送する。

⑥被制御処理装置2の内部コマンド処理部21は、レジスタリードライト処理部22を介し、プログラムカウンタにアドレスA番地をセットする。プログラムカウンタのセットによりアドレスA番地実行後、アドレス(A+1)番地を実行するとプログラム割込みが発生し、被制御処理装置2の割込み処理部24に制御が渡る。割込み処理部24は、レジスタを退避する等の割込み処理を行い、内部コマンド処理部21に制御を渡す。内部コマンド処理部21は、割込みが発生したことを、レジスタ等の割込み情報を付加し制御処理装置1に転送する。

⑦制御処理装置1のステップコマンド処理部13は、被制御処理装置2からの情報で、割込みが期待した割込みであることを確認すると、レジスタ内容や更に必要な情報があれば被制御処理装置2に転送を要求する等し、その後、確保していたアドレス(A+1)番地の内容をメモリライトコマンド指定の内部コマンドで被制御処理装置2に戻して、1ステップ実行処理を終える。

【0014】⑧ステップコマンド処理部13は、コマンド翻訳処理部10、外部コマンド切替処理部9、オペレータ入出力サポート処理部8を介してオペレータに表示した後、オペレータの確認を待つ。オペレータが確認し次の1ステップの実行を続ける場合、次のアドレスを指定して同様の処理を繰り返す。なお、上述の説明では、実行命令が分岐命令でないことを想定して説明しているが、分岐命令のサポートは被制御処理装置2の命令によ

るため、被制御処理装置2で吸収する。例えば、前記①の実行に先立ち、ステップコマンド処理部はA番地の内容をチェックする命令チェック処理部の内部コマンドを転送し、被制御処理装置2の内部コマンド処理部21は、命令チェック処理部25で分岐命令かどうか判定し、当該命令の直後のアドレスと分岐先アドレス、例えばB番地を、転送する。ステップコマンド処理部13は、実行命令の直後と分岐先の両方の内容を確認し、両方に、プログラム割込みを起こす命令、共通の戻り先である内部コマンド処理部21に分岐する命令等を書き込む。

【0015】また、前記プログラム割込みを起こす命令は被制御処理装置2で用意し、制御処理装置1からは被制御処理装置2の命令実行の流れを変える命令を書き込む内部コマンドを転送する。ここでは、説明を簡単にするため、命令の長さは1番地分であつ分岐命令ではないことを前提としている。また、外部コマンドファイル17からの自動デバッグを行う場合は、予め外部コマンドをパラメータ形式デバッグ手順通りに外部コマンドファイル17に格納しておき、例えば、制御処理装置デバ

ッガ5のロード時、自動デバッグモードをロードパラメータで与える等して、ロード後、外部コマンド切替処理部9は、前記モードを特定エリア等から読み取り自動デバッグモードとする。外部コマンド切替処理部9は、オペレータ入出力サポート処理部8に外部コマンドを要求する代わりに、外部コマンドロード処理部11に外部コマンドを要求する。外部コマンドロード処理部11は、外部コマンドファイル17から外部コマンドを順次読み出し、外部コマンド切替処理部9に渡す。

【0016】外部コマンド切替処理部9は、オペレータ入出力サポート処理部8からの入力と同様に、コマンド翻訳処理部10に外部コマンドを渡す。外部コマンドの実行終了後に、コマンド翻訳処理部10から外部コマンドの実行結果が送られてくると、外部コマンド切替処理部9は、実行結果の情報をデバッグ結果ログ処理部12に渡し、デバッグ結果ログ処理部12は、これをデバッグ結果ログファイル18に格納する。以上のように、オペレータが指示する場合と同様にデバッグを自動的に実行できる。オペレータはすべてのデバッグが完了した後、デバッグ結果ログファイル18をチェックし、デバ

10

20

30

40

# 【0017】

【発明の効果】以上、詳細に説明した如く、本発明によれば、異なった種類毎の被制御処理装置に対応して同一の制御処理装置のデバッガが使用でき、また、デバッグするプログラムが実際の装置上で動作することで処理速度の変動が無いリモートデバッグ方法を実現できるという顕著な効果を奏するものである。より詳細には、被制御処理装置のアーキテクチャに関わる固有の部分のみ被制御処理装置デバッガで専用に吸収することにより、制御処理装置デバッガに汎用性を持たせることができ、また、機能追加等は制御処理装置デバッガのみに施せば、本デバッグシステム全体に適応でき、拡張性も高い。更に、外部コマンドファイルに外部コマンドをパラメータ化してデバッグ手順通りに格納しておくことにより、デバッグの自動化が計れる。また、外部コマンドファイルに外部コマンドのパラメータを累積することにより、常に、以前デバッグを行った項目を繰返し実行できるため、プログラムのデグレードを防止し、信頼度を維持できる。また、被制御処理装置の装置番号を外部コマンドに設定することにより、複数台のアーキテクチャの異なった被制御処理装置の異なったプログラムのデバッグが自動的に実行できる。

# 【0018】

## 【図面の簡単な説明】

【図1】本発明の一実施例を示すリモートデバッガのプログラムブロック図の一部である。

【図2】本発明の一実施例を示すリモートデバッガのプログラムブロック図の他の一部である。

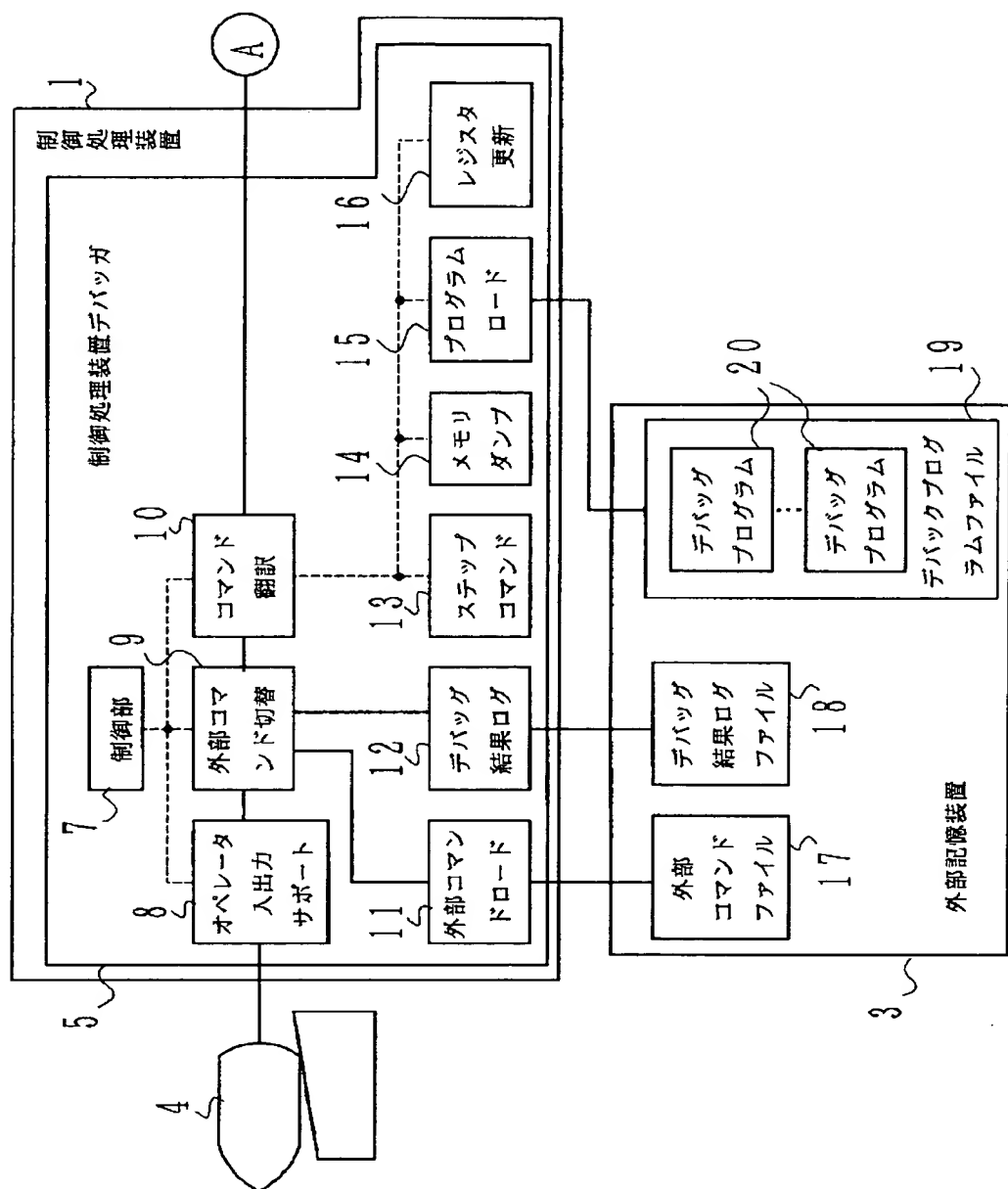
【図3】図1中のコマンド翻訳処理部10のプログラムブロック図である。

## 【符号の説明】

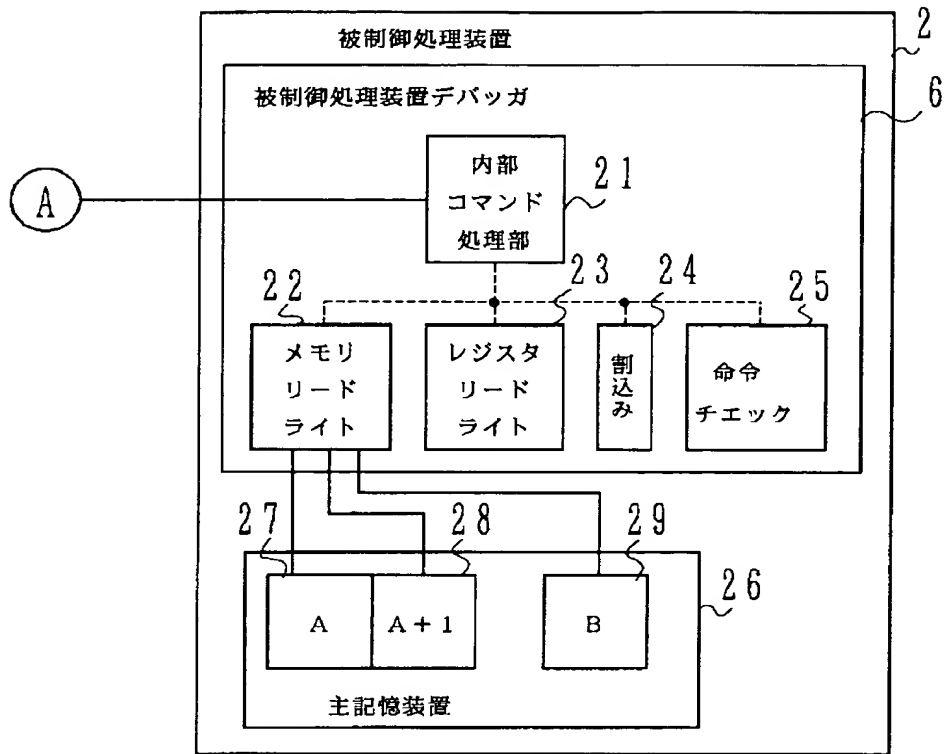
1：制御処理装置、2：被制御処理装置、3：制御処理装置の外部記憶装置、4：オペレータ入出力装置、5：制御処理装置デバッガ、6：被制御処理装置デバッガ、7：制御部、8：オペレータ入出力サポート処理部、9：外部コマンド切替処理部、10：コマンド翻訳処理部、11：外部コマンドロード処理部、12：デバッグ結果ログ処理部、13：ステップコマンド処理部、14：メモリダンプ処理部、15：プログラムロード処理部、16：レジスタ更新処理部、17：外部コマンドファイル、18：デバッグ結果ログファイル、19：デバッグプログラムファイル、20：デバッグプログラム、21：内部コマンド処理部、22：メモリリードライト処理部、23：レジスタリードライト処理部、24：割込み処理部、25：命令チェック処理部、26：主記憶装置、27：アドレスA、28：アドレス(A+1)、29：アドレスB。



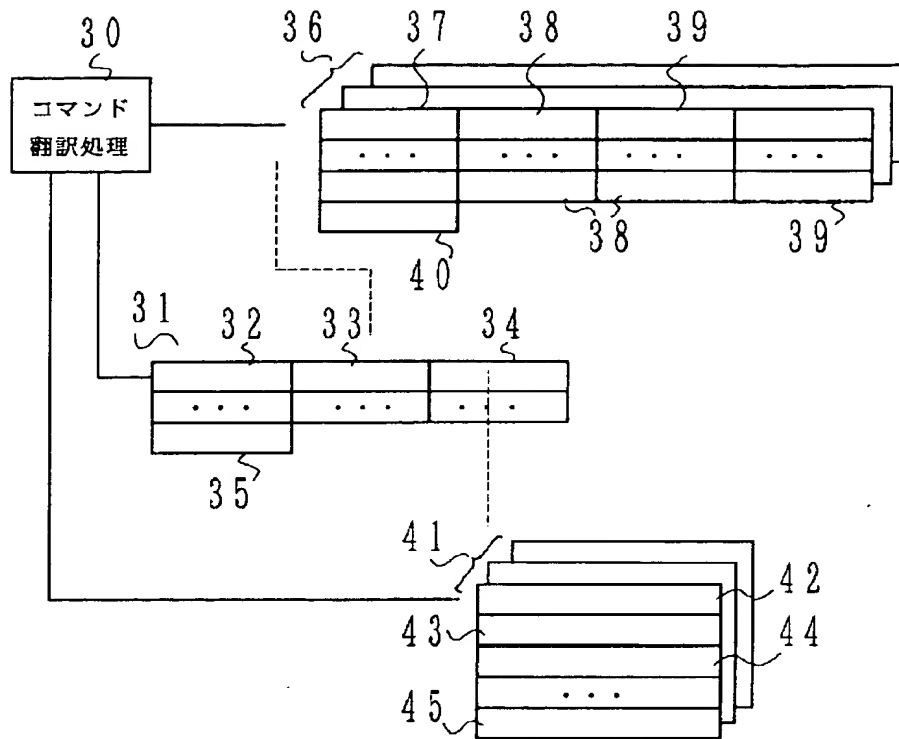
【図1】



【図2】



【図3】



フロントページの続き

(72)発明者 萩原 利夫  
愛知県尾張旭市晴丘町池上1番地 株式会  
社日立製作所旭工場内

(72)発明者 高木 利彰  
愛知県尾張旭市晴丘町池上1番地 株式会  
社日立製作所旭工場内

(72)発明者 田村 英克  
愛知県尾張旭市晴丘町池上1番地 株式会  
社日立旭エレクトロニクス内